## REMARKS

Claims 2, 4, 7 and 8 are amended as suggested by the examiner to overcome the informalities. Claim 2 is also put in independent form. Claim 8 has been amended to be in independent form. It is assumed that the formal drawings are approved since they have been reviewed and there has been no objection.

Claims 1-3 are rejected under 35 U.S.C. 103 (a) as being unpatentable over Gheith ( U.S. Patent No. 5,797,014 ; hereinafter Gheith) , in view of Long et al ( U.S. Patent No. 5,835,958; hereinafter Long).

The Gheith reference is directed to a process for compiling shared library code to minimize global offset table address computation. This reference establishes a link convention requiring that each link module have a single location containing a pointer to the global offset table (GOT) for that module. Each exported function in the module is pre-appended with a word containing the offset of the GOT pointer from the function entry point. In the preferred embodiment, the word immediately preceding the function entry point contains the offset to the GOT pointer.

Claim 1, as amended, calls for "determining if a transfer of control is beyond a near call limitation and if so generating a link time modification of object code by the compiler or assembler by the addition of custom generated object code or trampoline code to the link without changing the compiler generated instructions or expanding compiler generated object code for a long distance transfer of control by redirecting original call to a code which will transfer control to the original target address." There is no suggestion of this in Gheith. There is no trampoline code for long distance transfer of control by redirecting the original call to a code which will transfer control to the original target address in Gheith.

The examiner references Long at Column 7 line 46 through Column 8, line 36. The Long reference describes a method for efficiently allocating discontinuous stack space without requiring compiler changes. The method includes calling a stack checking function that includes the compiled function. A determination is made if additional memory is required for executing the compiled function. If no additional memory is required, then the function is called and executed. If additional memory is necessary, the additional memory is allocated that is discontinuous with the original memory stack. The examiner references a section discussing calling a trampoline function. The trampoline function of the reference is associated with trampoline function frame 318 that is invoked by the buffer frame to release the scheduler lock to enable the actual function, which is associated with actual function frame 322, to be invoked. See bottom of Column 7 and top of column 8. The trampoline function of the reference has a prologue that contains locking code or code which is used to lock the stack protection lock and an epilogue that contains code which is used to unlock the stack protection lock. See Column 8, lines 10 thru 17. It does not discuss or teach "determining if a transfer of control is beyond a near call limitation and if so generating a link time modification of object code by the compiler or assembler by the addition of custom generated object code or trampoline code to the link without changing the compiler generated instructions or expanding compiler generated object code for a long distance transfer of control by redirecting original call to a code which will transfer control to the original target address." The trampoline function relates to a stack locking code. It is not seen how this relates to determining if a transfer control is too far for a near call and if too far to add to the link a custom generated object code for a long distance transfer of control by redirecting the original call to a code which will transfer control to the target address. There is no determination if a transfer is too far. There is no teaching of if too far of providing for a long distance transfer of control by redirecting the original call to a code that will transfer control to the target address. There is no teaching of adding to a link such a custom generated object code. Claim 1 is therefore deemed allowable over the references.

Claim 2 is amended to be in independent form and is deemed allowable for at least the same reasons as Claim 1. Claim 2 further calls for "using a sequence of trampolines." It is not seen where this is in any way suggested in any of the references.

The examiner suggestion that this would have been obvious is without any support in the references. The examiner suggests that there would be motivation being that the more run time re-allocation is required to effect remotely addressable instructions, the more the need to establish prologue and epilogue code. It is not seen where the examiner gets such motivation or there is a motivation in the need for a sequence of trampolines to establish prologue and epilogue. There is no motivation discussed or suggested by the references for a sequence of trampolines. The examiner's suggestion of motivation is totally without merit. Applicant's teach beginning on page 24, line 20 that a single trampoline may not work in certain conditions such as the need to use resources such as registers to prepare for a far call; the need to use different entry points for near and far calls; or other conditions in which a far call to the original call target cannot be made in a way that mimics a near call from the original caller. Only applicant teaches using a sequence of trampolines or a motivation for a sequence of trampolines.

Furthermore, in regard to combining the cited prior art, reference is made to In re Fritch, 23 USPQ2d 1780 and particularly the portion thereof at page 1783 under "Prima Facie Obviousness" where the Court stated:

> "In proceedings before the Patent and Trademark Office, the Examiner bears the burden of establishing a prima facie case of obviousness based upon the prior art. '[The Examiner] can satisfy this burden only by showing some objective teaching in the prior art or that knowledge generally available to one of ordinary skill in the art would lead that individual to combine the relevant teachings of the references.' The patent applicant may then attack the Examiner's prima facie determination as

improperly made out, or the applicant may present objective evidence tending to support a conclusion of nonobviousness.'"

and later stated:

"'Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination. Under section 103, teachings of references can be combined only if there is some suggestion or incentive to do so.' Although couched in terms of combining teachings found in the prior art, the same inquiry must be carried out in the context of a purported obvious 'modification' of the prior art. The mere fact that the prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggested the desirability of the modification."

There is no suggestion in the references to suggest the combination claimed or the desirability of the modification.

In view of the above applicants claim 2, as amended, is deemed allowable over the references.

Claim 3 calls for "A method of making far calls or branches comprising the step of providing link time modification of object code generated by the compiler or assembler by the addition of custom generated object code to the link without changing the compiler generated instructions or expanding compiler generated object code." It is not seen where the cited references teach the method of making far calls by the link time modification of object code by the addition of custom generated object code as claimed for the reasons discussed above.

Applicants Claim 4–9 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gheith and Long and further in view of Kurahashi (U.S. Patent No. 5,740,447; hereinafter Kurahashi).

The Kurahashi reference discusses a branch instruction optimizing process which performs optimization of code length of branch instructions for branching between modules. As discussed in the background of the invention on page 12 of applicants' specification, the Kurahashi patent call for the linker to change near branches to far branches and vice versa. Depending on the computer architecture, this method may require the linker to substitute different branch instructions of possibly different sizes, and possibly insert additional instructions. Requiring additional instructions is notable because some computer architectures require the use of a register in order to load an address for a far branch. Therefore, when a near branch is produced by the compiler, a free register may not be available to use for the new far branch inserted by the linker. Further it may be necessary to change near-return instruction to far return instructions if the original branch is actually a subroutine call instruction. Link time fix-up as described in the reference may be difficult to implement in a linker because changing a branch may cause other changes to ripple through the object file as its size changes and secondly the modified object code may be less efficient. Kurahashi likewise does not teach or suggest "creating a second trampoline S2 and modifying S1 to perform a far call to B2 in S2 and add any necessary setup code to S2 and subroutine B2 in S2 is made to contain a near call to target T and return to consider the next call" as claimed in claim 4. There is nothing in any of the references to teach or suggest this. The whole methodology is not taught by Kurahashi, Gheith and Long. The optimizing processing portion includes instruction a judging portion for making judgment of branch instructions, etc. but teaches nothing about trampoline code or judgment of branch instructions with respect to trampoline code. There is no teaching or suggestion of "for each near external call C, the linker computing the distance from C to its target T and performing the following steps: determining if the call C and target T are allocated close enough to each other to permit a near call and if so, then near call C target T directly with no modification and return to consider the next call; otherwise if there is there already a trampoline S1 to target T that is linkably close enough to call C to permit a near call, then modifying call C to point to call B1 in S1 and returning to consider the next call; other wise creating a second trampoline S2 and modifying S1 to perform a far call to call B2 in S2 and add any necessary setup code to S2 and subroutine call B2 in S2 is made to contain a near call to target T and return to consider the next call." There is nothing in either of Gheith, Long or Kurahashi or their combination to teach this. There is no teaching of when there is a near call and when there is a far call or that there be creating a second trampoline and modifying the first trampoline S1 to perform a far call to B2 and setup code to call B2 in S2 and subroutine call B2 is made to contain a new call to target T and return to consider the next call in the second trampoline S2 and add step code to the second trampoline a near call. There is no teaching or motivation for any of this without a teaching of a sequence of trampolines. Applicant's Claim 4, as amended, is therefore deemed allowable over the references.

Claim 7 calls for "A method of fixing link time problems relating to out of range branch or call instructions comprising the steps of: computing if the target is too far distant from the branch or call; if it is too far distant then determining if there already is a trampoline section to the target and if so redirect the new call or branch to that trampoline section and if there is not already a trampoline to the target then generating a trampoline section to the target and redirect the near call or branch to the generated trampoline section." There is no such teaching in the combination Kurahashi, Gheith or Long for the reasons discussed previously.

Claim 8, as amended, calls for "if a single-trampoline fails to work because of resources, then included are the step of: generating a second trampoline and generating a far branch or call from said first trampoline to the second trampoline section and generating at the second trampoline section a near call or branch to the original target." As discussed previously nothing like this is taught or suggested in the cited references. There is no motivation found in the references for the second trampoline or how they may be coupled. There is no second trampoline and no suggestion of generating a far branch or call from the first trampoline to the second trampoline and/or generating at the second trampoline a near call to the original target. Claim 8, as amended, is therefore deemed allowable over the references.

Claim 9 dependent on Claim 8 is deemed allowable for at least the same reasons as Claim 8. Claim 9 further calls for "wherein the return is a near return from the target to the second trampoline, a far return from the second trampoline, and a near return from the first trampoline to the original call." There is no teaching in the references with respect to a second trampoline or how there is a near return from the target to the second trampoline, a far return from the second trampoline or a near return from the first trampoline to the original call.

In view of the above applicants' claims 1 thru 9, as amended, are deemed allowable and an early notice of allowance of these claims is deemed in order and is respectfully requested.

Respectfully submitted;

Robert L. Troike

Robert L. Troike (Reg. 24183)

Tel. No. 301-259-2089